Games for learning: a design pattern approach (Workshop)

Yishay Mor
yishaym@gmail.com

Niall Winters
n.winters@ioe.ac.uk

Knowledge Lab

The design of a technology enhanced learning (TEL) tool, and specifically of Interactive Learning Environments (ILEs), is a major challenge. This is because it must address issues ranging from learning theory to software engineering. Developers face fundamental challenges in building tools to adequately address the issues raised during the design process. However, understanding and resolving each of the requirements and the tensions between participants has long been recognized as fundamental to any tool's success. Mor & Winters (in press) present a thematic review of design approaches in TEL, highlight some of the key challenges and suggest that a design pattern approach may offer a way forward. The current workshop offers an innovative approach for designing technology-enhanced learning environments, with special attention to computer games. We will briefly present the theoretical underpinnings of our work, and position the design pattern approach in relation to several other constructs, such as design principles (Kali, 2006), Scriptlets (Schank) and Situated Abstraction (Noss & Hoyles, 1996). We then proceed to demonstrate our methods by working through selected examples. The workshop structure will encourage audience participation and active debate. We will conclude with a discussion on possible applications of the design patterns approach to participants' practice. We invite interested parties to use our tools and methods and develop their own pattern languages. While our initial focus has been on the design and use of games for mathematical learning, the community process we engendered led us to appreciate the parallels in a broader set of domains. We encourage participants to relate our examples to their own domains of expertise and identify which elements of design are common and which are unique.

The Kaleidoscope project Learning Patterns for the Design and Deployment of Games for Mathematical Learning (http://lp.noe-kaleidoscope.org) draws on traditions of design science and participatory design to offer an innovative approach for designing technology-enhanced learning environments. This approach extends Alexander's design patterns (Alexander, 1977) and supplements them with a set of bespoke on-line collaborative tools. These tools were used by a group of experts to develop a language of patterns for game-based mathematical learning. This language was refined through a series of workshops. We propose these workshops as a model for multi-disciplinary participatory design of educational technologies. Our language of patterns aims to formalize expert insights from the different domains of design knowledge. Our patterns are intended to serve the needs of researchers, practitioners and designers as one. The structure of these patterns is based on the long-standing tradition of pattern languages in various fields, adopted to the special focus of games and learning. We began with a process of collaborative reflection on our own experiences, and have expanded this process to include other researchers and practitioners through an on-going series of

workshops. Our language of patterns, and the collaborative on-line tool that support it, is evolving continuously as our knowledge grows. We do not claim to offer a comprehensive set of patterns, but we do strive to construct a coherent language, which has few holes and many open ends. Our aim is it to address issues across a broad range of aspects pertaining to the process of designing, implementing and deploying games for mathematical learning. The learning patterns we developed attempt to strike a balance between problem solving and being feature specific. Some patterns address the process of game development and in doing so emerged from problems we were trying to overcome. As such, they can be viewed as problem solving patterns. Others are directly concerned with particular game features and interaction issues and are considered feature specific. However, in describing the patterns we use the generic term 'problem' to encompass both perspectives. For example, to address a particular problem the designer may add a specific feature.

We work from the premise that designing and deploying technology learning is a difficult task because it requires the assimilation and integration of deep knowledge from diverse domains of expertise including content knowledge, interface and interaction engineering, software engineering, learning and teaching. We see all these aspects of knowledge as various facets of *design knowledge*. The content dimension pertains to the question of selecting and connecting elements of domain knowledge – a question of designing ontological structures. The question of pedagogy is a question of designing instructional structures, and so on. While each party may have expertise in several of the associated domains, no single party has expertise in all of them. The complexity of each of the various bodies of knowledge means that it is often hard to communicate ideas between parties. Each community has developed its own lore and jargon. The result of this fragmentation of knowledge is that most games emerge from a particular, often restricted viewpoint. For example, when developing mathematical games, a game that embodies deep mathematical can be poorly designed in terms of the gaming experience, whereas a sleek and entertaining game may be simplistic in its pedagogical intent.

Design approaches in technology-enhanced learning generally and Interactive Learning Environments in particular, are strongly influenced by the seminal work of Simon (1969), who was the first to refer to design as a science. Simon distinguishes between the natural sciences and the sciences of the artificial, challenging the view of the latter as 'practical' science or 'vocational arts'. At the core of the study of the artificial, Simon places the science of design. In his words, "*everyone designs who devises courses of action aimed at changing existing situations into desired ones*" (Simon, 1969, p 129). We identify three key elements in Simon's approach which guide our work:

- A prescriptive agenda. Whereas natural science is concerned with what *is*, design science asks what *ought to be*.
- Function as the axis of decomposition. Whereas natural science progresses by structural decomposition and synthesis, design science analyses its objects of study by their purpose and the aims they claim to serve.
- Centricity of representation. Our capacity to solve problems is contingent on the way these problems are represented.

We argue that *design patterns* (Alexander et al., 1977) hold a powerful promise for recording, calibrating and collaboratively refining expert knowledge. Patterns are flexible enough to address a very broad spectrum of practices, from in-depth technical development to deployment issues in classrooms. In addition, they are rigid enough to oblige the pattern writer to focus on and concisely capture their own best practice. The design patterns approach was developed as a form of design language within architecture. This was done with the explicit aim of externalizing knowledge to allow accumulation and generalization of solutions and to allow all members of a community or design group to participate in discussion relating to the design. A design pattern "describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice" (Alexander et al., 1977, p. X). Design patterns have the explicit aim of externalizing knowledge to allow accumulation and generalization of solutions and to allow all members of a community or design group to participate in discussion relating to the design. Patterns are organized into coherent systems called pattern languages where patterns are related to each other. Although the use of design patterns never achieved a large following among professional architects, the idea has been embraced in several other disciplines, including software engineering (Gamma et al., 1995), hypermedia (German & Cowan, 2000) and interaction design (Erickson, 2000; Borcher, 2001). The approach has also found application in educational domains including e-learning systems (Derntl & Motschnig-Pitrik, 2004) and the design of computer science courses (Bergin, 2000).

Our pattern language and its associated interactive tools are presented as resources to be used by researchers and practitioners in several ways. As an analytical asset, design patterns are a means of making visible implicit design decisions. Researchers and designers can reflect on their own work by mapping it to patterns in our language, or by extending the language to account for aspects we do not cover. Identifying the underlying patterns can help understand the strengths and weaknesses of existing games and the ways in which they are used. Once a pattern has been mapped to a case under observation, the context noted in the pattern can be compared to the details of the actual case, and conflicts can be discussed. On the other hand, the related patterns should be explored, to identify possible extensions and enhancements. As a design aid, practitioners from various fields who are involved in game design and deployment can consult the patterns in different stages of their process, and chose those which address the particular questions they are confronted with. Some of our patterns address the flow of the process as a whole, some address specific phases - such as 'bootstrapping' design, and some offer concrete structural elements which can by used as building blocks. It is important to note that patterns are not cookbooks. They do not devolve responsibility from the designer, but only help her understand the scope of the issues to consider and scaffold her attempts to resolve these.

However, the most important facet of the pattern language is its potential as a framework for discussing and collaboratively refining design. In fact, this is precisely why it is called a pattern *language*, and not collection or set. This language grew through its use in various assemblies of designers, researchers and educators. Our workshops are structured around the language and the tools, and have used them successfully to sustain effective communication among experts from varied

backgrounds. In this function, our pattern language should be seen as a starting point, an example - from which each community will derive and develop its own language. The process of creating, or 'distilling' a pattern begins with reflection on expert knowledge represented as a case study of good practice. The pattern authors identify a single element of design which contributed to the success of this case study. This element is phrased in a manner which detaches it from the single example, but avoids over-abstraction. The pattern is carefully named: names need to be descriptive, concise and attractive. Its details are then moulded into the pattern template. Once the pattern has been described, it is mapped to other case studies and to other patterns in the language. By comparing to similar case studies we can refine the pattern and identify its critical features. This may lead to the need to define new patterns - as special cases of this one or as generalizations of it. At the same time, we classify new pattern using the hierarchical structure of the language and look for related patterns which are already in our collection. The pattern language was developed iteratively and collaboratively by the project team, in dialogue with a wider community of designers, researchers and teachers. Due to the distributed nature of the team, the availability of on-line tools played an important role in our ability to conduct this process effectively. These tools were developed in parallel with the language, as our understanding of the process we were engaged in evolved.

Alongside the development of the pattern language, we have developed a set of interactive tools to support it. The primary functions of these tools are to allow us to efficiently manage the pattern language, and at the same time make it easy to use by any interested reader. These tools provide various methods of browsing, reading, editing and organizing patterns. The pattern browser is the central tool in our system. It provides several modes for viewing the patterns, as well as entry points to tools for creating new patterns and structuring the language. All patterns are listed in a database, and can be viewed in table mode and sorted by various keys. The hierarchical structure of the language is represented in a FreeMind map, which can be viewed and used as a navigation scheme for accessing pattern pages. Patterns are edited using an on-line rich text editor. This editor is based on an open source tool with slight modifications and enhancements. A pattern page begins with a header which displays an expanded view of the meta-data listed in the browser index view. Pattern pages are generated from a template, which scaffolds the author to use a common structure. This structure includes a concise statement of the pattern's intent or the problem it addresses, a detailed delineation of its context, a description of the pattern itself, its relations to other patterns, additional notes and examples. The notes will typically refer to underlying educational research. The examples point to the relevant case studies in our collection.

We see this as an exciting opportunity to reflect on our results in a critical environment, and explore opportunities for future collaborations.

References

Alexander, C., Ishikawa, S. and Silverstein, M. (1977) *A Pattern Language: Towns, Buildings, Construction* (Center for Environmental Structure Series), Oxford University Press, New York.

Bergin, J. (2000) *Fourteen Pedagogical Patterns*. Proceedings of the Fifth European Conference on Pattern Languages of Programs.

Borchers, J. (2001) *A Pattern Approach to Interaction Design*, John Wiley and Sons, Chichister, England.

Derntl, M. and Motschnig-Pitrik, R. (2004) *Patterns for blended, Person-Centered learning: strategy, concepts, experiences, and evaluation*. In Proceedings of ACM Symposium on Applied Computing (SAC), pp. 916-923.

Erickson, T. (2000) *Lingua Francas for design: sacred places and pattern languages*. In Proceedings of conference on Designing interactive systems. ACM Press, New York, pp. 357-368.

German, D. M. and Cowan, D. D. (2000) *Towards a unified catalog of hypermedia design* patterns. In Proceedings of the 33rd Annual Hawaii International Conference on System Sciences, pp. 6067-6075.

Kali, Y. (2006) *Collaborative knowledge building using a design principles database*. ijcscl, 1, 187-201.

Mor, Y. and Winters, N. (in press) Design approaches in technology enhanced learning. *Interactive Learning Environments*.